PACK CONTROLLER

M-CDFA002-01B

Ｅｔｈｅｒｎｅｔ　ＡＥ－ＬＩＮＫコンバータ

ＲｏＨＳ指令適合

Ｃ１６４０

＜取扱説明書　プログラム作成編＞

１．目次

２．はじめに

この度は弊社製品をご利用頂きまして、誠に有り難うございます。
本製品は小型ながら多くの機能・性能を備えております。その効果を有効かつ安全に活用して頂く為にも、
ご使用前に取扱説明書（本書）を必ずお読み下さい。
お読みになった後も、いつでも読めるように所定の場所に保管して下さい。

当製品は一般的な産業機器の組込用として設計・製造されています。医療用機器・原子力関係・その他
直接人命に関わる機器等には使用しないでください。また、本書の警告・注意事項等を守らなかった場合に
生じた損害の補償について、当社は一切その責任を負いませんので、あらかじめご了承下さい。

３．安全上の注意点

この取扱説明書では、安全注意事項のランクを『警告』『注意』と区分してあります。

⚠ 警告 ：取扱を誤った場合に、危険な状況が起こりえて、死亡または重傷を受ける可能性が想定される場合。

⚠ 注意 ：取扱を誤った場合に、危険な状況が起こりえて、中程度の傷害や軽傷を受ける可能性が想定される場合、および物的傷害のみの発生が想定される場合。

なお、⚠ 注意 に記載した事項でも、状況によっては重大な結果に結びつく可能性があります。

いずれも重要な内容を記載していますので必ず守って下さい。

⚠ 警告

全般

○爆発性雰囲気、引火性ガスの雰囲気、腐食性の雰囲気・水・油、その他液体のかかる場所、可燃物のそばでは使用しないで下さい。火災、怪我の原因になります。
○通電状態で移動、取り付け、接続、点検の作業を行わないで下さい。必ず電源を切ってから作業して下さい。怪我、コントローラ破損の原因になります。
○取り付け・接続・点検等の作業は、機器の知識、安全の情報そして注意事項に習熟した人が行って下さい。火災、怪我、コントローラ破損の原因になります。

接続

○コントローラの電源入力電圧は、定格範囲を必ず守って下さい。火災、コントローラ破損の原因となります。
○接続は接続図に基づき確実に行って下さい。火災、コントローラ破損の原因となります。
○電源を投入した状態での接続は絶対に行わないで下さい。感電、火災、装置破損の恐れがあります。
○電源線や信号線を無理に曲げる、引っ張る、はさみ込む等行わないで下さい。
火災、コントローラ破損の原因となります。

修理・分解・改造

○修理・分解・改造は行わないで下さい。怪我・火災・その他重大な結果に結びつく可能性があります。
○接続作業は、機器の知識、安全の情報そして注意事項に習熟した人が行って下さい。

⚠ 注意

全般
○コントローラの仕様値を超えての使用はしないで下さい。怪我、装置破損の原因となります。
○通電中や電源遮断後しばらくの間は、コントローラ・モータが熱くなっている場合がありますので、
触れないで下さい。怪我の原因となります。

保管
○雨や水滴のかかる場所・有害なガスや液体のある場所には保管しないで下さい。
コントローラ破損の原因となります。
○日光の直接当たらない場所で、決められた湿度・温度範囲で保管して下さい。
コントローラ破損の原因となります。

設置
○周囲温度が５０℃を越えるようなときは、ファン等で強制冷却し、表面温度が６０℃以下になるように
してください。やけど・火災・装置破損の恐れがあります。
○コントローラに重いものをのせたり、乗ったりしないでください。怪我、コントローラ破損の恐れがあります。
○金属などの不燃物に取り付けてください。火災の恐れがあります。
○コントローラと制御板の内面または、その他の機器との間隔は規定の距離を保ってください。
火災の原因となります。

運転
○機械系と結合し試運転を行う場合は、いつでも非常停止できる状態にしてから行って下さい。
怪我、装置破損の原因となります。
○装置の故障や動作異常が発生したときは、装置全体が安全な方向に働くよう非常停止装置、または
非常停止回路を外部に設置して下さい。怪我の原因になります。
○アラームが発生した場合は直ちに運転を停止して、コントローラの電源を遮断して下さい。
そのまま運転を続けると火災、怪我の原因になります。
○運転中は駆動部分に触れないでください。巻き込まれ、怪我の原因になります。
○製品の内蔵スイッチは絶縁されたマイナスドライバ等を使用してください。また、スイッチの設定は
電源ＯＦＦ状態で行って下さい。怪我、コントローラ破損の原因になります。

保守・点検
○通電中・電源切断直後はコントローラに触れないで下さい。怪我の原因になります。
○絶縁抵抗・絶縁耐圧試験の際は、端子に触れないで下さい。怪我の原因になります。

廃棄
○コントローラを破棄する場合は産業廃棄物として処理して下さい。

セキュリティ
○本機器はグローバルＩＰが設置されることは考慮に入れて設計されていません。情報漏洩の恐れがあります。
○プロトコルの仕様上、暗号化していないデータのやり取りがございます。不特定多数のアクセスのあるネットワークに
設置しないでください。情報漏洩の恐れがあります。

４．マニュアルの構成

マニュアル名称	記述内容
C1640 取扱説明書	C1640 に関する全般的な説明とハードウエア的な接続、設置から接続確認に至るまでの手順を記載しています。
C1640 取扱説明書プログラム作成編	ソフトウエア的なインターフェースに関する記述をしています。

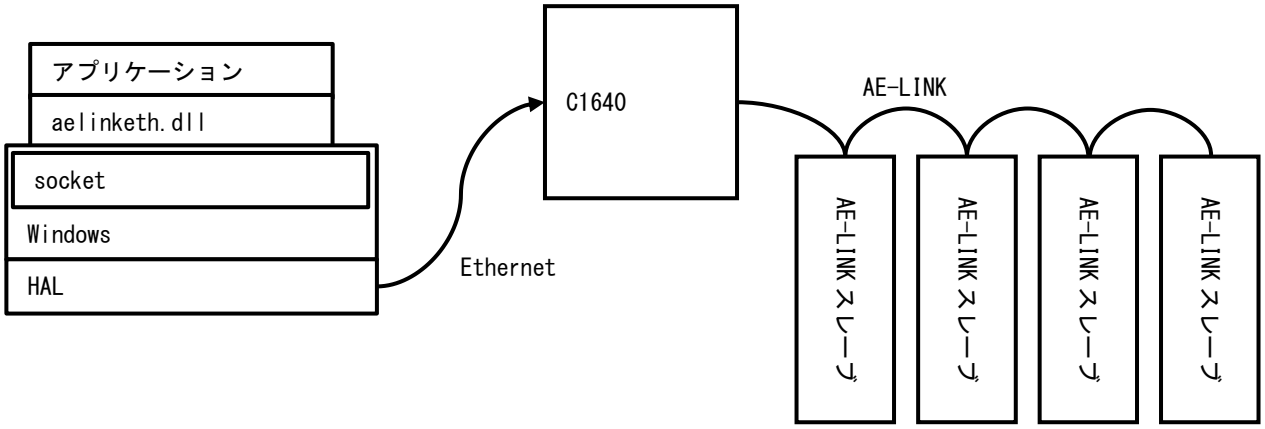
５．本マニュアルで対象とするＯＳとプログラム言語

対応 OS	Windows 10 x64 Windows 10 Windows 8.1 x64 Windows 8.1 Windows 8 x64 Windows 8 Windows 7 x64 Windows 7 Windows Vista x64 Windows Vista Windows XP x64 Windows XP
対象言語	Microsoft Visual C++ (.NET2002, .NET2003, 2005 ～ 2017) Microsoft Visual C# (.NET2002, .NET2003, 2005 ～ 2017) Microsoft Visual Basic (.NET2002, .NET2003, 2005 ～ 2017)

６．システムの構成について

Windows 環境下で C1640 を使用するためのアクセス用のライブラリとして「aelinketh.dll」を使用してください。
「aelinketh.dll」では Windows に内臓の socket 通信ライブラリを使用して、C1640 に通信を行います。
Windows の設定によってはファイアウォールの条件に該当する場合は所定の設定を行ってください。

＜概念図＞



７．制御開始までの手続き

制御開始するまでのステップは以下の通りです。
※ソフトウエア的な手順を示してあります。ハード的な手続きについては「C1640 取扱説明書」をご確認ください。

ステップ１	**設置**	８．設置
	ファイルの配置を行います。	

⇩

ステップ２	**C1640,DLL 初期化**	９．C1640,DLL 初期化
	プログラムで DLL の初期化を行います。	

⇩

ステップ３	**スレーブ初期設定**	１０．スレーブ初期設定
	各スレーブの初期化を行います。	

⇩

ステップ４	**自動送信スレーブの登録**	１１．自動送信スレーブの登録
	自動的に状態を読み出すためのスレーブ登録を行います。	

⇩

ステップ５	**原点復帰**	１２．原点復帰
	スレーブに対して原点復帰動作を行うことで、物理的な位置を担保します。	

⇩

ステップ６	**運用動作**	１３．運用動作
	目的の動作を行います。	

⇩

ステップ7	終了処理	１４．終了処理
	終了のための処理を行います。	

８．設置

８－１．ビット数による使用ファイルの違いについて

Windows のビット数によってダイナミックライブラリや関連ファイルの名称が異なります。
開発環境に合わせて設置や設定を行ってください。

使用言語	Windows ビット数	アプリケーションビット数	ファイル	備考
C,C++	32bit	32bit	aelinketh.dll aelinketh.exp aelinketh.lib aelinketh.h(共通)	aelinketh.lib プロジェクトプロパティで [リンカー]-[入力]- [追加の依存ファイル]に追加
	64bit	32bit	aelinketh.dll aelinketh.exp aelinketh.lib aelinketh.h(共通)	aelinketh.lib プロジェクトプロパティで [リンカー]-[入力]- [追加の依存ファイル]に追加
		64bit	aelinketh64.dll aelinketh64.exp aelinketh64.lib aelinketh.h(共通)	aelinketh64.lib プロジェクトプロパティで [リンカー]-[入力]- [追加の依存ファイル]に追加
C#	32bit,64bit	32bit,64bit	aelinketh.cs(共通)	
VB	32bit,64bit	32bit,64bit	aelinketh.vb(共通)	

８－２．DLL の設置場所について

Windows の仕様に従い下記の場所の何れかに設置してください。

１、実行中のプロセスの実行形式モジュールがあるフォルダー。(推奨)
２、現在のフォルダー このフォルダーへのパスは、GetCurrentDirectory 関数で取得します。
３、Windows システム フォルダー。 このフォルダーへのパスは、GetSystemDirectory 関数で取得します。
４、Windows ディレクトリ。 このフォルダーへのパスは、GetWindowsDirectory 関数で取得します。
５、環境変数 PATH 内に記述されたフォルダー。

９．C1640,DLL 初期化

ハードウエア的な設置や設定方法については、「C1641 取扱説明書」に従って C1640 の設定を行ってください。
他ライブラリと同じように本ライブラリにおいても、必ずソフトウエアを使用する前に AelEth_Open()で初期化する必要があります。初期化を行うとライブラリを運用するためのリソースの確保と、自動送信確認用のスレッドを作成します。
AelEth_Open()を行うにあたっては、下記のパラメータを指定する必要があります。

- [Prm.0-0]自局 IP アドレス（※C1640 のスイッチが加算されるので注意）
- [Prm.0-4] AE-LINK 通信 UDP ポート番号

例：自局 IP アドレス=192.168.0.250、AE-LINK 通信 UDP ポート番号=6000 の場合

[C,C++]
```
AELINKETH aeleth;
// 通信用コネクションの確保
aeleth = AelEth_Open("192.168.0.250", 6000);
if (aeleth == NULL)
{
   // open error
}
```
[C#]
```
using static AelinkEthAPI.aelinketh;
IntPtr aeleth;
// 通信用コネクションの確保
aeleth = aelinketh.AelEth_Open("192.168.0.250", 6000);
if (aeleth == IntPtr.Zero)
{
    // open error
}
```

[VB]
```
Dim aeleth As IntPtr
' 通信用コネクションの確保
aeleth = Aelinketh.AelEth_Open("192.168.0.250", 6000)
If aeleth = IntPtr.Zero Then
    ' open error
End If
```

１０．スレーブ初期設定

スレーブの設定方法に従って、最初にパラメータの設定を行って下さい。
個々のスレーブの仕様に従って AE-LINK コマンド発行してください。

旭エンジニアリング製のドライバは下記のようなパラメータがあります。
基本的には各パラメータは初期値を持っているので、そのままで問題無い場合にはコマンドを発行する必要はありません。
また、旭エンジニアリング製のドライバは全てのパラメータ設定が完了した後、
必ずシステム設定完了コマンド（３９ｈ）を発行して下さい。

設定内容	コマンド番号	コード例
システム設定完了コマンド	３９ｈ	[C,C++] AelEth_SimpleSet(addr,0x39,0,0); [C#] AelEth_SimpleSet(addr,0x39,0,0); [VB] Aelinketh.AelEth_SimpleSet(addr,&H39,0,0)

ａ．動作パターンに関わらず設定が必要なパラメータ

設定内容	コマンド番号	コード例
動作時電流値設定	３０ｈ	[C,C++] AelEth_SimpleSet(addr,0x30,設定値,1); [C#] AelEth_SimpleSet(addr,0x30,設定値,1); [VB] Aelinketh.AelEth_SimpleSet(addr,&H30,設定値,1)
停止時電流値設定	３１ｈ	[C,C++] AelEth_SimpleSet(addr,0x31,設定値,1); [C#] AelEth_SimpleSet(addr,0x31,設定値,1); [VB] Aelinketh.AelEth_SimpleSet(addr,&H31,設定値,1)
加速時電流値設定	３２ｈ	[C,C++] AelEth_SimpleSet(addr,0x32,設定値,1); [C#] AelEth_SimpleSet(addr,0x32,設定値,1); [VB] Aelinketh.AelEth_SimpleSet(addr,&H32,設定値,1)
動作パターン設定	３４ｈ	[C,C++] AelEth_SimpleSet(addr,0x34,設定値,1); [C#] AelEth_SimpleSet(addr,0x34,設定値,1); [VB] Aelinketh.AelEth_SimpleSet(addr,&H34,設定値,1)
回転方向設定	３Ａｈ	[C,C++] AelEth_SimpleSet(addr,0x3A,設定値,1); [C#] AelEth_SimpleSet(addr,0x3A,設定値,1); [VB] Aelinketh.AelEth_SimpleSet(addr,&H3A,設定値,1)
現在位置設定	２９ｈ	[C,C++] AelEth_SimpleSet(addr,0x29, 位置,4); [C#] AelEth_SimpleSet(addr,0x29, 位置,4); [VB] Aelinketh.AelEth_SimpleSet(addr,&H29, 位置,4)

※スレーブによって設定の有無や、単位系が異なる場合がございます。
　設定値に付いては個々のドライバの仕様をご確認ください。

ｂ．原点復帰動作を行う場合に設定が必要なパラメータ

設定内容	コマンド番号	コード例
原点復帰 高速周波数	２５ｈ	[C,C++] AelEth_SimpleSet(addr,0x25,設定値,2); [C#] AelEth_SimpleSet(addr,0x25,設定値,2); [VB] Aelinketh.AelEth_SimpleSet(addr,&H25,設定値,2)
原点復帰 起動周波数	２６ｈ	[C,C++] AelEth_SimpleSet(addr,0x26,設定値,2); [C#] AelEth_SimpleSet(addr,0x26,設定値,2); [VB] Aelinketh.AelEth_SimpleSet(addr,&H26,設定値,2)
加減速率	２３ｈ	[C,C++] AelEth_SimpleSet(addr,0x23,設定値,2); [C#] AelEth_SimpleSet(addr,0x23,設定値,2); [VB] Aelinketh.AelEth_SimpleSet(addr,&H23,設定値,2)
原点復帰 サーチ周波数	２７ｈ	[C,C++] AelEth_SimpleSet(addr,0x27,設定値,2); [C#] AelEth_SimpleSet(addr,0x27,設定値,2); [VB] Aelinketh.AelEth_SimpleSet(addr,&H27,設定値,2)
原点復帰 最大時間設定	２８ｈ	[C,C++] AelEth_SimpleSet(addr,0x28,設定値,1); [C#] AelEth_SimpleSet(addr,0x28,設定値,1); [VB] Aelinketh.AelEth_SimpleSet(addr,&H28,設定値,1)
原点復帰 動作方法設定	３６ｈ	[C,C++] AelEth_SimpleSet(addr,0x36,設定値,2); [C#] AelEth_SimpleSet(addr,0x36,設定値,2); [VB] Aelinketh.AelEth_SimpleSet(addr,&H36,設定値,2)

※スレーブによって設定の有無や、単位系が異なる場合がございます。
設定値に付いては個々のドライバの仕様をご確認ください。

ｃ．連続回転／絶対位置決め／相対位置決め動作を行う場合に設定が必要なパラメータ

設定内容	コマンド番号	コード例
高速周波数	２１ｈ	[C,C++] AelEth_SimpleSet(addr,0x21,設定値,2); [C#] AelEth_SimpleSet(addr,0x21,設定値,2); [VB] Aelinketh.AelEth_SimpleSet(addr,&H21,設定値,2)
起動周波数	２２ｈ	[C,C++] AelEth_SimpleSet(addr,0x22,設定値,2); [C#] AelEth_SimpleSet(addr,0x22,設定値,2); [VB] Aelinketh.AelEth_SimpleSet(addr,&H22,設定値,2)
加減速率	２３ｈ	[C,C++] AelEth_SimpleSet(addr,0x23,設定値,2); [C#] AelEth_SimpleSet(addr,0x23,設定値,2); [VB] Aelinketh.AelEth_SimpleSet(addr,&H23,設定値,2)

※スレーブによって設定の有無や、単位系が異なる場合がございます。
設定値に付いては個々のドライバの仕様をご確認ください。

ｄ．脱調検出を行う場合に設定が必要なパラメータ

設定内容	コマンド番号	コード例
脱調検出 機能設定	３Ｄｈ	[C,C++] AelEth_SimpleSet(addr,0x3D,設定値,1); [C#] AelEth_SimpleSet(addr,0x3D,設定値,1); [VB] Aelinketh.AelEth_SimpleSet(addr,&H3D,設定値,1)
脱調検出幅設定	３Ｅｈ	[C,C++] AelEth_SimpleSet(addr,0x3E,設定値,2); [C#] AelEth_SimpleSet(addr,0x3E,設定値,2); [VB] Aelinketh.AelEth_SimpleSet(addr,&H3E,設定値,2)
エンコーダ 分解能設定	３Ｆｈ	[C,C++] AelEth_SimpleSet(addr,0x3F,設定値,4); [C#] AelEth_SimpleSet(addr,0x3F,設定値,4); [VB] Aelinketh.AelEth_SimpleSet(addr,&H3F,設定値,4)

※スレーブによって設定の有無や、単位系が異なる場合がございます。
設定値に付いては個々のドライバの仕様をご確認ください。

１１．自動送信スレーブの登録

自動送信の設定を行います。
自動送信設定を行わない場合や設定の仕方によって、状態読み出しを行う AelEth_Position(),AelEth_EncPosition(),
AelEth_Status(),AelEth_IsMoveing(),AelEth_IsAlram()は不定状態になります。使用する機能に従って設定してください。
自動送信設定後、AelEth_AutoStart()で自動送信を開始してください。

[C,C++]
```
// 自動送信の設定
AelEth_AutoAEDrvEntry(aeleth, 0x00, AETH_AUTO_STATUS_POS);
AelEth_AutoAEDrvEntry(aeleth, 0x01, AETH_AUTO_STATUS_POS);

// 自動送信スタート
AelEth_AutoStart(aeleth);
```

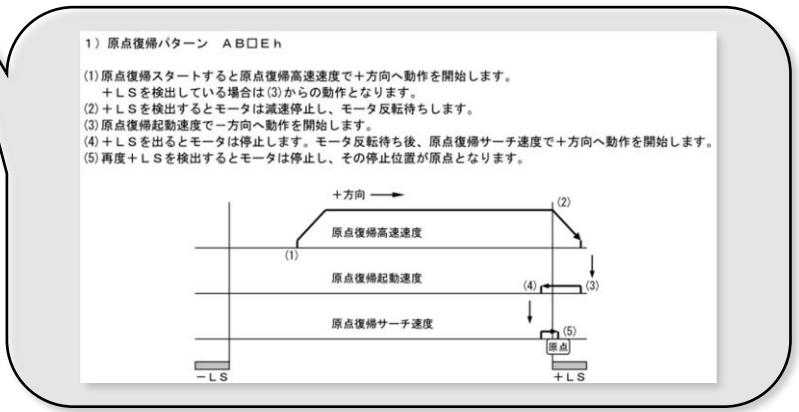
[C#]
```
using static AelinkEthAPI.aelinketh;
// 自動送信の設定
AelEth_AutoAEDrvEntry(aeleth, 0x00, AethAutoSelect.AETH_AUTO_STATUS_POS);
AelEth_AutoAEDrvEntry(aeleth, 0x01, AethAutoSelect.AETH_AUTO_STATUS_POS);

// 自動送信スタート
AelEth_AutoStart(aeleth);
```

[VB]
```
' 自動送信の設定
Aelinketh.AelEth_AutoAEDrvEntry(aeleth, &H0, AethAutoSelect.AETH_AUTO_STATUS_POS)
Aelinketh.AelEth_AutoAEDrvEntry(aeleth, &H1, AethAutoSelect.AETH_AUTO_STATUS_POS)

' 自動送信スタート
Aelinketh.AelEth_AutoStart(aeleth)
```

１２．原点復帰

原点復帰を行うスレーブに対しては原点復帰を行ってください。
原点復帰処理のスレーブ指令は、下記のような方法を推奨します。
１、原点復帰関連パラメータの送信
２、原点復帰動作開始指令
３、原点復帰完了待ち
４、機械原点まで位置決め動作
５、機械原点移動待ち
６、現在位置設定コマンドで０に設定

[C,C++]
```
// 原点復帰関連パラメータ設定
AelEth_SimpleSet(aeleth, addr, 0x23, 50, 2);          // 加減速率設定
AelEth_SimpleSet(aeleth, addr, 0x25, 500, 2);         // 原点復帰高速速度設定
AelEth_SimpleSet(aeleth, addr, 0x26, 10, 2);          // 原点復帰起動速度設定
AelEth_SimpleSet(aeleth, addr, 0x27, 10, 2);          // 原点復帰サーチ速度設定
AelEth_SimpleSet(aeleth, addr, 0x28, 100, 1);         // 原点復帰最大時間設定
AelEth_SimpleSet(aeleth, 0, 0x36, 0xAB4E, 2);         // 原点復帰パターン

// 原点復帰開始
if(AelEth_Homing(aeleth, 0) != AETH_SUCCESS)
{
    // error
}


// 停止待ち
while (AelEth_IsMoveing(aeleth, 0)) { }


// 機械原点へ位置決め
if(AelEth_Absolute(aeleth, addr, 1000) != AETH_SUCCESS)
{
    // error
}

// 停止待ち
while (AelEth_IsMoveing(aeleth, 0)) { }

// 現在位置に0設定
AelEth_SimpleSend(aeleth, 0, 0x29, 0, 4);
```



[C#]
```
// 原点復帰関連パラメータ設定
AelEth_SimpleSet(aeleth, addr, 0x23, 50, 2);          // 加減速率設定
AelEth_SimpleSet(aeleth, addr, 0x25, 500, 2);         // 原点復帰高速速度設定
AelEth_SimpleSet(aeleth, addr, 0x26, 10, 2);          // 原点復帰起動速度設定
AelEth_SimpleSet(aeleth, addr, 0x27, 10, 2);          // 原点復帰サーチ速度設定
AelEth_SimpleSet(aeleth, addr, 0x28, 100, 1);         // 原点復帰最大時間設定
AelEth_SimpleSet(aeleth, addr, 0x36, 0xAB4E, 2);      // 原点復帰パターン

// 原点復帰開始
if (AelEth_Homing(aeleth, addr) != AethResult.AETH_SUCCESS)
{
    // error
}
```

```
// 停止待ち
while (AelEth_IsMoveing(aeleth, addr)) { }


// 機械原点へ位置決め
if (AelEth_Absolute(aeleth, addr, 1000) != AethResult.AETH_SUCCESS)
{
    // error
}

// 停止待ち
while (AelEth_IsMoveing(aeleth, addr)) { }

// 現在位置に0設定
AelEth_SimpleSend(aeleth, addr, 0x29, 0, 4);

[VB]
‘ 原点復帰関連パラメータ設定
Aelinketh.AelEth_SimpleSet(aeleth, addr, &H23, 50, 2)           ’ 加減速率設定
Aelinketh.AelEth_SimpleSet(aeleth, addr, &H25, 500, 2)          ’ 原点復帰高速速度設定
Aelinketh.AelEth_SimpleSet(aeleth, addr, &H26, 10, 2)           ’ 原点復帰起動速度設定
Aelinketh.AelEth_SimpleSet(aeleth, addr, &H27, 10, 2)           ’ 原点復帰サーチ速度設定
Aelinketh.AelEth_SimpleSet(aeleth, addr, &H28, 100, 1)          ’ 原点復帰最大時間設定
Aelinketh.AelEth_SimpleSet(aeleth, addr, &H36, &HAB4E, 2)       ’ 原点復帰パターン

’ 原点復帰開始
If (Aelinketh.AelEth_Homing(aeleth, addr) <> AethResult.AETH_SUCCESS) Then
    ’ Error
End If


’ 停止待ち
While (Aelinketh.AelEth_IsMoveing(aeleth, addr))
End While


’ 機械原点へ位置決め
If (Aelinketh.AelEth_Absolute(aeleth, addr, 1000) <> AethResult.AETH_SUCCESS) Then
    ’ error
End If


’ 停止待ち
While (Aelinketh.AelEth_IsMoveing(aeleth, addr))

End While

’ 現在位置に0設定
Aelinketh.AelEth_SimpleSend(aeleth, addr, &H29, 0, 4)
```

１３．運用動作

■通信ステータス、機器ステータス状態の読み出し
AelEth_Status()で確認することが出来ます。

■異常状態のについて
AelEth_IsAlram()で確認することが出来ますので、異常と判定した場合、詳細を確認することを推奨いたします。

```
[C,C++]
if(AelEth_IsAlram(aeleth,addr))
{
   alarm = AelEth_SimpleGet(aeleth,addr,0x42,0,1);
}
[C#]
if(AelEth_IsAlram(aeleth,addr))
{
   alarm = AelEth_SimpleGet(aeleth,addr,0x42,0,1);
}
 [VB]
If Aelinketh.AelEth_IsAlram(aeleth, addr) Then
    alarm = Aelinketh.AelEth_SimpleGet(aeleth,addr,0x42,0,1)
End If
```

■位置の状態の読み出し
AelEth_Position ()で確認することが出来ます。

■エンコーダ位置の読み出し
AelEth_EncPosition ()で確認することが出来ます。

■動作の実行について
AelEth_Jog(),AelEth_Homing(),AelEth_Relative(),AelEth_Absolute()にて動作開始を行うことが出来ます。
動作中の判定はAelEth_IsMoveing()で確認することが出来ます。

１４．終了処理

アプリケーション終了時には必ず、AelEth_Close()を行って終了処理を行ってくだい。
※自動送信読み出し用のスレッドが残ってしまい、アプリケーションが終了できない場合があります。

```
[C,C++]
AelEth_Close(aeleth);
```

```
[C#]
aelinketh.AelEth_Close (aeleth);
```

```
[VB]
Aelinketh.AelEth_Close (aeleth)
```

１５．リファレンス

１５－１．エラーコード一覧

値	定義	説明
0	AETH_SUCCESS	正常終了
1	AETH_AELINK_NOTEXEC_CMDERR	AE-LINK コマンドとしては送受信できたが、命令実行不可、コマンドエラーとなった。
-1	AETH_NG	異常終了
-2	AETH_ERR_SEND	Ethernet UDP 送信異常
-3	AETH_ERR_RECVTIMEOUT	Ethernet UDP 受信タイムアウト送信後 500ms 待っても受信しなかった場合
-4	AETH_ERR_RECVSIZE	Ethernet UDP 受信サイズ異常
-5	AETH_ERR_FORMAT	Ethernet UDP 受信フォーマット
-6	AETH_ERR_AELINK	AELINK 通信異常
-7	AETH_ERR_SIZEOVER	関数引に異常値を設定した場合

１５－２．関数一覧

区分	関数名	説明
共通	AelEth_Open	初期化処理を行います。
	AelEth_Close	終了処理を行います。
	AelEth_Send	AE-LINK 通信を行います。
	AelEth_SimpleComm	定型の AE-LINK 通信を行います。
	AelEth_SimpleSet	設定用の定型の AE-LINK 通信を行います。
	AelEth_SimpleGet	取得用の定型の AE-LINK 通信を行います。
	AelEth_MakePacket	AE-LINK パケットを生成します。
自動送信	AelEth_AutoStart	自動送信の実行を指令します。
	AelEth_AutoStop	自動送信の停止を指令します。
	AelEth_AutoAEDrvEntry	旭エンジニアリング製ドライバの自動送信設定を行います。
	AelEth_AutoExEntry	自動送信設定を行います。
	AelEth_GetAutoData	自動送信で取得したデータを読み出します。
	AelEth_IsAutoReading	自動送信中であることを確認します。
拡張	AelEth_Initialize	スレーブに対してイニシャライズ指令を行います。
	AelEth_Reset	スレーブに対してリセット指令を行います。
	AelEth_Jog	スレーブに対して連続回転指令を行います。
	AelEth_Homing	スレーブに対して原点復帰指令を行います。
	AelEth_Relative	スレーブに対して相対位置決め指令を行います。
	AelEth_Absolute	スレーブに対して絶対位置決め指令を行います。
	AelEth_DStop	スレーブに対して減速停止指令を行います。
	AelEth_QStop	スレーブに対して即停止指令を行います。
自動送信読み	AelEth_Position	スレーブに対して自動送信データを元に現在位置を取得します。
	AelEth_EncPosition	自動送信データを元にエンコーダ位置を取得します。
	AelEth_Status	自動送信データを元にステータスを取得します。
	AelEth_IsMoveing	自動送信データを元に動作状態を判定します。
	AelEth_IsAlram	自動送信データを元にアラーム状態を判定します。

１６．共通処理関数

１６－１．AelEth_Open()

機能		初期化処理を行います。
宣言	C,C++	AELINKETH AELINKETH_API AelEth_Open(const char *target_ip, UINT16 target_port);
	C#	public static System.IntPtr AelEth_Open(string target_ip, UInt16 target_port);
	VB.NET	Public Shared Function AelEth_Open(ByVal target_ip As String, ByVal target_port As UInt16) As IntPtr
引数	target_ip [入力]	C1640 の IP アドレスを指定してください。
	target_port [入力]	C1640 のポート番号を指定してください。
戻り値		成功の場合：初期化済みの新しいハンドルが返ります。 失敗の場合：NULL

解説

アプリケーションを作成する際は、必ず本処理で初期化を行ってください。
複数の C1640 に接続する際は、接続数分、初期化を行う必要があります。

１６－２．AelEth_Close()

機能		終了処理を行います。
宣言	C,C++	void AELINKETH_API AelEth_Close(AELINKETH ael);
	C#	public static void AelEth_Close(System.IntPtr ael);
	VB.NET	Public Shared Sub AelEth_Close(ByVal ael As IntPtr)
引数	ael[入力]	終了処理を行いたいハンドル
戻り値		なし

解説

アプリケーションを終了する際は、必ず本処理で終了処理を行ってください。
複数の C1640 に接続する際は、対応するハンドル分を行う必要があります。

１６－３．AelEth_Send()

機能		AE-LINK 通信を行います。
宣言	C,C++	AELINKETH_STATUS AELINKETH_API AelEth_Send(AELINKETH ael, const BYTE* cmd, UINT32 cmd_size, BYTE* recv, UINT32 recv_size);
	C#	public static Int32 AelEth_Send(System.IntPtr ael, byte[] cmdbff, out byte[] recvbff);
	VB.NET	Public Shared Function AelEth_Send(ByVal ael As IntPtr, ByVal cmdbff As Byte(), ByRef recvbff As Byte()) As Int32
引数	ael[入力]	初期化済みハンドル
	cmdbff[入力]	送信 AE-LINK データの先頭ポインタを指定します。パケット長とサムチェック値は関数内で自動的付加して AE-LINK データとして C1640 に送信します。
	cmd_size [入力]	cmdbff で示す AE-LINK 送信データのサイズを指定します。 ※C#,VB の場合、cmdbff.Length で確認します。
	recvbff [出力]	AE-LINK 通信が正常に完了した場合に受信した AE-LINK 応答データを指定したバッファに書き込みます。NULL を指定した場合も正常に動作しますが、応答データは破棄されます。
	recv_size [入力]	AE-LINK 応答データサイズ： recvbff で指定したバッファのサイズを指定してください。 ※C#,VB の場合、自動判定します。
戻り値		正常に完了した場合、受信したバイト数を示します。 なお、AE-LINK の通信ステータス、「命令実行不可」、又は「コマンドエラーの場合」は AETH_AELINK_NOTEXEC_CMDERR が返ります。 それ以外の、異常の場合はエラーコード一覧を参照してください。

解説

指定された、コマンドを AE-LINK 通信として発信します。
なお、C1640 との通信中は制御を返しません。

使用例

一般例

コマンド	コマンド名	通信方向	DATA 数 [byte]	データ範囲	単位	説明
0x00	リセット	発行	0	—	—	ドライバアラームのリセットを行います。
		応答	0	—	—	

コマンドの並び

送信パケット	byte 数	応答パケット
0x04+発行 DATA 数	0	0x04+応答 DATA 数
アドレス	1	アドレス
コマンド	2	通信ステータス
データ （データ数は コマンドによって異なります）		データ （データ数は コマンドによって異なります）
サムチェック		サムチェック

指定の DATA 数だけデータが入ります。
特に指定のない限り
リトルエンディアンでやり取りを行います

サムチェック以外の各バイトを加算した値の
下位バイトをサムチェック値とします。

AelEth_Send(AELINKETH ael, const BYTE* cmd, UINT32 cmd_size, BYTE* recv, UINT32 recv_size);

例１
送信するスレーブのアドレスが「0x00」で速度＝10.0rpm を設定する場合

コマンド	コマンド名	通信方向	DATA 数 [byte]	データ範囲	単位	説明
0x53	速度指令	発行	4	-4000.0rpm～ 4000.0rpm	0.1rpm	～～～～～～～
		応答	0	―	―	

コマンドの並び

送信パケット	byte 数	応答パケット
0x08	0	0x04
0x00	1	0x00
0x53	2	通信ステータス
0x64	3	サムチェック
0x00	4	
0x00	5	
0x00	6	
0xBD	7	

※通信ステータスはスレーブの状態によって変化します。

```
BYTE cmdbff[6] = {0x00,0x53,0x64,0x00,0x00,0x00};
BYTE recvbff[0x100];

AELINKETH_STATUS sts;
sts = AelEth_Send(ael, cmdbff, 6, recvbff,sizeof(recvbff));
if(sts >= AETH_AELINK_NOTEXEC_CMDERR)
{
    // error
}
```

例２
送信するスレーブのアドレスが「0x10」で速度読み出しコマンドを送信した場合

コマンド	コマンド名	通信方向	DATA 数 [byte]	データ範囲	単位	説明
0x46	速度読出し	発行	0	―	―	現在の速度を応答します。
		応答	4	-10000.0 ～ 10000.0	0.1rpm	

コマンドの並び

送信パケット	byte 数	応答パケット
0x04	0	0x04
0x10	1	0x10
0x46	2	通信ステータス
0x54	3	0x**
	4	0x**
	5	0x**
	6	0x**
	7	サムチェック

※通信ステータスはスレーブの状態によって変化します。

基本的に送信データ、受信データのデータ並びは
リトルエンディアンになります。
例（※マイナスは２の補数表現します。）
４バイトの例
0x18 0xFC 0xFF 0xFF → -1000rpm
0x10 0x27 0x00 0x00 → 10000rpm
0xF0 0xD8 0xFF 0xFF → -10000rpm

```
BYTE cmdbff[2] = {0x10,0x46};
BYTE recvbff[0x100];

AELINKETH_STATUS sts;
sts = AelEth_Send(ael, cmdbff, 2, recvbff,sizeof(recvbff));
if(sts >= AETH_AELINK_NOTEXEC_CMDERR)
{
    // error
}
int spd = 0;
spd |= (((int)recvbff[3]) << 0 ) & 0x000000FF;
spd |= (((int)recvbff[4]) << 8 ) & 0x0000FF00;
spd |= (((int)recvbff[5]) << 16) & 0x00FF0000;
spd |= (((int)recvbff[6]) << 24) & 0xFF000000;
```

１６－４．AelEth_SimpleSet()

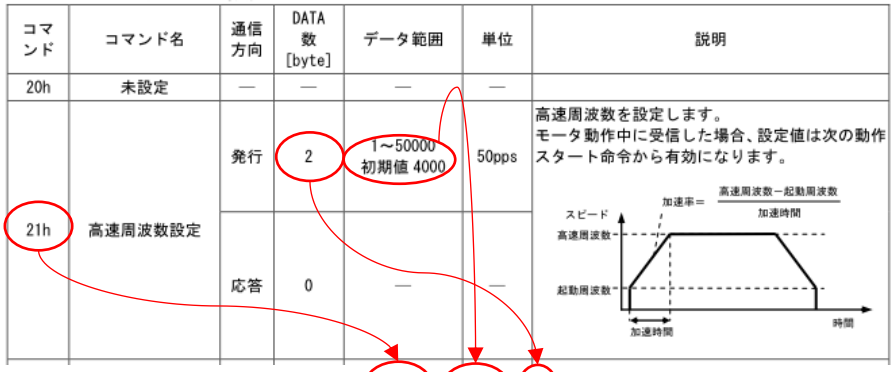
機能		定型の設定用 AE-LINK 通信を行います。
宣言	C,C++	AELINKETH_STATUS AELINKETH_API AelEth_SimpleSet(AELINKETH ael, BYTE addr, BYTE cmd, INT32 val, BYTE size);
	C#	public static Int32 AelEth_SimpleSet(System.IntPtr ael, byte addr, byte cmd, Int32 val, byte size);
	VB.NET	Public Shared Function AelEth_SimpleSet(ByVal ael As System.IntPtr, ByVal addr As Byte, ByVal cmd As Byte, ByVal val As Int32, ByVal size As Byte) As Int32
引数	ael	初期化済みハンドル：
	addr	送信対象 AE-LINK アドレス
	cmd	送信コマンド番号
	val	送信データ
	size	送信データサイズ
戻り値		正常に完了した場合、受信したバイト数を示します。 なお、AE-LINK の通信ステータス、「命令実行不可」、又は「コマンドエラーの場合」は AETH_AELINK_NOTEXEC_CMDERR が返ります。 それ以外の、異常の場合はエラーコード一覧を参照してください。

解説

addr で指定した AE-LINK スレーブに対して、定型のコマンドを送信します。
本関数では、受信データは取得できません。
なお、C1640 との通信中は制御を返しません。

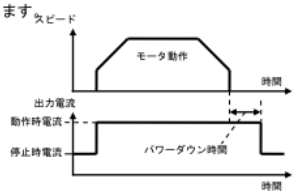
使用例

・AE-LINK アドレス=1 に対して、下記のコマンド 21h に 1234 を設定する例




```
if(AelEth_SimpleSet(ael, 0x01, 0x21, 1234, 2) >= AETH_AELINK_NOTEXEC_CMDERR)
{
}
```

・AE-LINK アドレス=5 に対して、下記のコマンド 30h に 100 を設定する例

コマンド	コマンド名	通信方向	DATA数 [byte]	データ範囲	単位	説明
30h	動作時電流設定	発行	1	0～255 初期値 50	0.01A	モータ動作中の出力電流を設定します。 モータ動作中に受信した場合、出力電流を即座に変更します。 
		応答	0	—	—	

```
if(AelEth_SimpleSet(ael, 0x05, 0x30, 100, 1) >= AETH_AELINK_NOTEXEC_CMDERR)
{
}
```

１６－５．AelEth_SimpleGet()

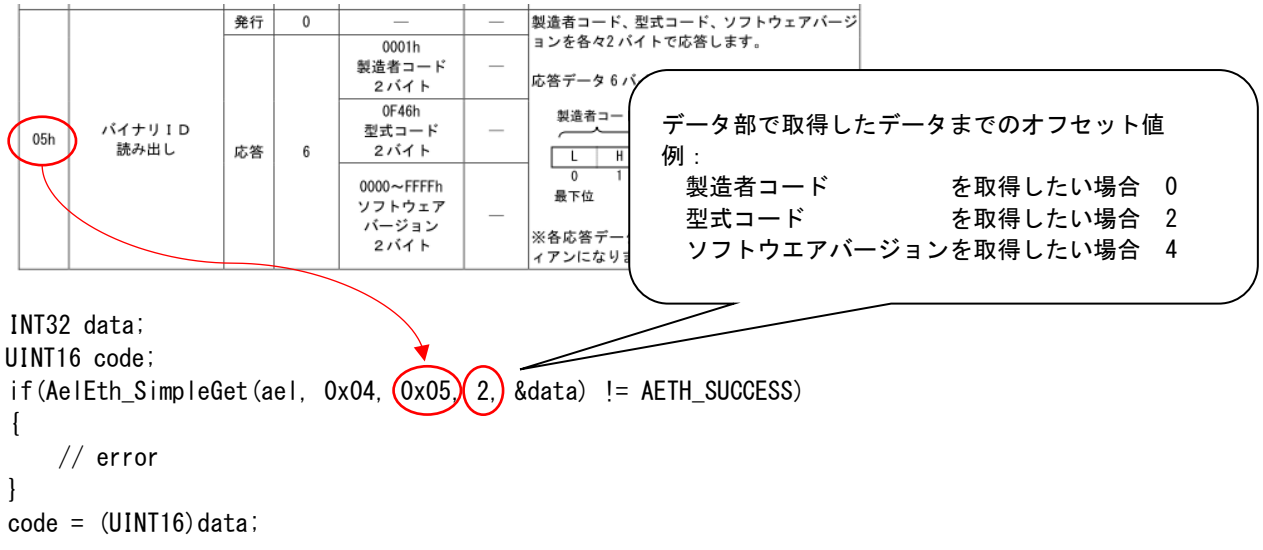
機能		定型のデータ取得用 AE-LINK 通信を行います。
宣言	C,C++	AELINKETH_STATUS AELINKETH_API AelEth_SimpleGet(AELINKETH ael, BYTE addr, BYTE cmd, BYTE offset, INT32 *recv_val);
	C#	public static Int32 AelEth_SimpleGet(System.IntPtr ael, byte addr, byte cmd, byte offset, out Int32 recv_val);
	VB.NET	Public Shared Function AelEth_SimpleGet(ByVal ael As System.IntPtr, ByVal addr As Byte, ByVal cmd As Byte, ByVal offset As Byte, ByRef recv_val As Int32) As Int32
引数	ael	初期化済みハンドル：
	addr	送信対象 AE-LINK アドレス
	cmd	送信コマンド番号
	val	送信データ
	val_size	送信データサイズ
	offset	応答データ取得オフセット
	recv_val	応答データ
戻り値		正常に完了した場合、AETH_SUCCESS が返ります。 異常の場合はエラーコード一覧を参照してください。

解説

addr で指定した AE-LINK スレーブに対して、定型のコマンドを送信します。
本関数では、受信データのバイト配列を数値に変換して返します。
なお、C1640 との通信中は制御を返しません。

使用例

・AE-LINK アドレス=4 に対して、下記のコマンド 05h で型式コードを取得する例




```
INT32 data;
UINT16 code;
if(AelEth_SimpleGet(ael, 0x04, 0x05, 2, &data) != AETH_SUCCESS)
{
    // error
}
code = (UINT16)data;
```

・AE-LINK アドレス=1 に対して、下記のコマンド 41h を取得する例

41h	現在スピード読み出し	発行	0	—	—	現在スピードを応答します。
		応答	2	1～65535	50pps	応答データに方向はありません。 モータ停止中の場合には 0 を応答します。

```
INT32 data;
UINT16 spd;
if(AelEth_SimpleGet(ael, 0x01, 0x41, 0, &data) != AETH_SUCCESS)
{
    // error
}
spd = (UINT16)data;
```

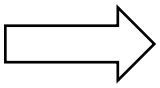
１６－６．AelEth_MakePacket()

機能		パケット長とサムチェック値が付いていない AE-LINK パケットを、パケット長とサムチェック値付加し、正規の AE-LINK パケットに変換します。
宣言	C,C++	AELINKETH_STATUS AELINKETH_API AelEth_MakePacket(BYTE* dest_fullpkt, UINT32 dest_size, const BYTE* src_basepkt, UINT32 src_size);
	C#	public static Int32 AelEth_MakePacket(byte[] dest_fullpkt, UInt32 dest_size, byte[] src_basepkt, UInt32 src_size);
	VB.NET	Public Shared Function AelEth_MakePacket(ByVal dest_fullpkt() As Byte, ByVal dest_size As UInt32, ByVal src_basepkt() As Byte, ByVal src_size As UInt32) As Int32
引数	dest_fullpkt	変換後の完成パケット保存先の先頭ポインタ：
	dest_size	パケット保存先のバッファサイズ：
	src_basepkt	変換前のパケット保存先の先頭ポインタ：
	src_size	変換前のサイズ：
戻り値		正常に完了した場合、AETH_SUCCESS が返ります。 異常の場合はエラーコード一覧を参照してください。

解説

正規の形に変換を行います。

なお、dest_fullpkt と src_basepkt に同じポインタを指定した場合は正しく動作しません。

src_basepkt	AelEth_MakePacket()処理後	dest_fullpkt
0x10		0x04
0x46		0x10
		0x46
		0x54

１７．自動送信処理関数

１７－１．AelEth_AutoStart()

機能		C1640 に対して自動送信の開始指令を行います。
宣言	C,C++	AELINKETH_STATUS AELINKETH_API AelEth_AutoStart(AELINKETH ael);
	C#	public static AethResult AelEth_AutoStart(System.IntPtr ael)
	VB.NET	Public Shared Function AelEth_AutoStart(ByVal ael As System.IntPtr) As Int32
引数	ael	初期化済みハンドル：
戻り値		正常に完了した場合、AETH_SUCCESS が返ります。 異常の場合はエラーコード一覧を参照してください。

解説

C1640 に対して自動送信制御コマンド５０を送信し開始指令を行います。
C1640 は電源投入時には自動送信機能は停止状態となっています。
本関数を処理することで自動送信を開始します。
本関数を行う前に、AelEth_AutoAEDrvEntry()や AelEth_AutoExEntry()を処理し、自動送信の設定を行ってください。
また、すでに開始指定している状態で、更に本関数を処理した場合の戻り値は正常となります。

１７－２．AelEth_AutoStop()

機能		C1640 に対して自動送信の停止指令を行います。
宣言	C,C++	AELINKETH_STATUS AELINKETH_API AelEth_AutoStop (AELINKETH ael);
	C#	public static AethResult AelEth_AutoStop(System.IntPtr ael)
	VB.NET	Public Shared Function AelEth_AutoStop(ByVal ael As System.IntPtr) As Int32
引数	ael	初期化済みハンドル：
戻り値		正常に完了した場合、AETH_SUCCESS が返ります。 異常の場合はエラーコード一覧を参照してください。

解説

C1640 に対して自動送信制御コマンド５０を送信し停止指令を行います。
また、すでに停止している状態で、更に本関数を処理した場合の戻り値は正常となります。

１７－３．AelEth_AutoAEDrvEntry()

機能		旭エンジニアリング社製の機器に対して自動送信設定を行います。
宣言	C,C++	AELINKETH_STATUS AELINKETH_API AelEth_AutoAEDrvEntry(AELINKETH ael, BYTE addr, int option);
	C#	public static AethResult AelEth_AutoAEDrvEntry(System.IntPtr ael, byte addr, AethAutoSelect option)
	VB.NET	Public Shared Function AelEth_AutoAEDrvEntry(ByVal ael As System.IntPtr, ByVal addr As Byte, ByVal op As AethAutoSelect) As Int32
引数	ael	初期化済みハンドル：
	addr	対象の AE-LINK アドレス
	option	下記の値を組み合わせで指定できます。 AETH_AUTO_STATUS AETH_AUTO_POSITION AETH_AUTO_STATUS_POS AETH_AUTO_ENCPOSITION
戻り値		正常に完了した場合、AETH_SUCCESS が返ります。 異常の場合はエラーコード一覧を参照してください。

解説

旭エンジニアリング製ドライバの自動送信設定を行います。
本関数を実行すると、対象のスレーブにバイナリーID 読み出しコマンドＯ５を送信し、旭エンジニアリング製のスレーブであることを確認します。確認後、option で指定されたコマンドを AelEth_AutoExEntry()を使用して設定を行います。
本関数は設定のみで、AelEth_AutoStart()を行うことで自動送信を開始します。

option の指定の仕方で、自動送信の登録コマンドが異なり、読み出すことできるデータが異なります。
option と自動送信設定の関係は下記の通りです。

option	対応関数	登録コマンド	想定応答と出力箇所
AETH_AUTO_STATUS	AelEth_Status()	0x04 addr 0x02 サムチェック	パケ長 アドレス 通信ステータス 機器ステータス サムチェック AelEth_Status()
AETH_AUTO_POSITION	AelEth_Position()	0x04 addr 0x40 サムチェック	パケ長 アドレス 通信ステータス 位置 LL 位置 LH 位置 HL 位置 HH サムチェック AelEth_Position()
AETH_AUTO_STATUS_POS	AelEth_Status() AelEth_Position()	0x04 addr 0x03 サムチェック	パケ長 アドレス 通信ステータス 機器ステータス 位置 LL 位置 LH 位置 HL 位置 HH サムチェック AelEth_Status() AelEth_Position()
AETH_AUTO_ENCPOSITION	AelEth_EncPosition()	0x04 addr 0x45 サムチェック	パケ長 アドレス 通信ステータス エンコーダ位置 LL エンコーダ位置 LH エンコーダ位置 HL エンコーダ位置 HH サムチェック AelEth_EncPosition()

１７－４．AelEth_AutoExEntry()

機能		汎用自動送信設定を行います。
宣言	C,C++	AELINKETH_STATUS AELINKETH_API AelEth_AutoExEntry(AELINKETH ael, BYTE option, BYTE recv_size, BYTE recv_offset, BYTE *send_cmd, BYTE cmd_size);
	C#	public static Int32 AelEth_AutoExEntry(System.IntPtr ael, byte option, byte recv_size, byte recv_offset, byte[] send_cmd);
	VB.NET	Public Shared Function AelEth_AutoExEntry(ByVal ael As System.IntPtr, ByVal op As Byte, ByVal recv_size As Byte, ByVal recv_offset As Byte, ByVal send_cmd()) As Int32
引数	ael	初期化済みハンドル：
	option	自動送信オプションを設定します C1640 コマンド 30 に従ってください。
	recv_size	1～16 の範囲で受信データの切り取りサイズを設定してください。
	recv_offset	応答パケットの先頭を 0 として、データコピー元を指定します。
	send_cmd	送信 AE-LINK データの先頭ポインタを指定します。パケット長とサムチェック値は関数内で自動的付加して AE-LINK データとして C1640 に送信します。
	cmd_size	送信 AE-LINK データのサイズを指定します。
戻り値		正常に完了した場合、C1640 自動送信 No が返ります。 異常の場合はエラーコード一覧を参照してください。

解説

C1640 に対して自動送信設定コマンド３０を使用して、任意コマンドの自動送信設定を行います。
自動送信の受信データは AelEth_GetAutoData()を使用して取得してください。

１７－５．AelEth_GetAutoData()

機能		自動送信応答データを取得します。
宣言	C,C++	AELINKETH_STATUS AELINKETH_API AelEth_GetAutoData(AELINKETH ael, BYTE entry, BYTE *buffer, BYTE buffer_size);
	C#	public static Int32 AelEth_GetAutoData(System.IntPtr ael, byte entry, out byte [] buffer);
	VB.NET	Public Shared Function AelEth_AutoExEntry(ByVal ael As System.IntPtr, ByVal entry As Byte, ByRef buffer() As Byte) As Int32
引数	ael	初期化済みハンドル
	entry	取得したい自動送信 No を指定してください。
	buffer	データコピー先のポインタを指定してください。
	buffer_size	buffer のサイズを指定してください。
戻り値		正常に完了した場合、C1640 自動送信 No が返ります。 異常の場合はエラーコード一覧を参照してください。

解説

自動送信データを読み出します。
事前に AelEth_AutoExEntry()で登録していない場合や、AelEth_AutoStart()で自動送信を開始していない場合、失敗します。

１７－６．AelEth_IsAutoReading()

機能		自動送信中であるかを確認します。
宣言	C,C++	BOOL AELINKETH_API AelEth_IsAutoReading (AELINKETH ael);
	C#	public static Int32 AelEth_GetAutoData(System.IntPtr ael);
	VB.NET	Public Shared Function AelEth_AutoExEntry(ByVal ael As System.IntPtr) As Int32
引数	ael	初期化済みハンドル
戻り値		実行中に完了した場合、TRUE が返します。 それ以外は、FALSE を返します。

解説

自動送信中であるかを確認します。

１８．拡張関数

１８－１．AelEth_Initialize()

機能		スレーブに対してイニシャライズ指令を行います。
宣言	C,C++	AELINKETH_STATUS AELINKETH_API AelEth_Initialize(AELINKETH ael, BYTE addr);
	C#	public static AethResult AelEth_Initialize(System.IntPtr ael, byte addr);
	VB.NET	Public Shared Function AelEth_Initialize(ByVal ael As System.IntPtr, ByVal addr As Byte) As Int32
引数	ael	初期化済みハンドル
	addr	対象の AE-LINK アドレス
戻り値		正常に完了した場合、AETH_SUCCESS が返ります。 異常の場合はエラーコード一覧を参照してください。

解説

addr で指定スレーブに対して、コマンド 01h を送信します。
スレーブのイニシャライズを指令する予約コマンドですが、指令後の挙動についてはスレーブに依存します。
送信 AE-LINK コマンドは下記の通りです

byte 数	送信パケット
0	0x04
1	addr
2	0x01
3	サムチェック

１８－２．AelEth_Reset()

機能		スレーブに対してリセット指令を行います。
宣言	C,C++	AELINKETH_STATUS AELINKETH_API AelEth_Reset(AELINKETH ael, BYTE addr);
	C#	public static AethResult AelEth_Reset(System.IntPtr ael, byte addr);
	VB.NET	Public Shared Function AelEth_Reset(ByVal ael As System.IntPtr, ByVal addr As Byte) As Int32
引数	ael	初期化済みハンドル
	addr	対象の AE-LINK アドレス
戻り値		正常に完了した場合、AETH_SUCCESS が返ります。 異常の場合はエラーコード一覧を参照してください。

解説

addr で指定スレーブに対して、コマンド 00h を送信します。
スレーブのリセットを指令する予約コマンドですが、指令後の挙動についてはスレーブに依存します。
送信 AE-LINK コマンドは下記の通りです

byte 数	送信パケット
0	0x04
1	addr
2	0x00
3	サムチェック

１８－３．AelEth_Jog()

機能		スレーブに対して連続回転指令を行います。
宣言	C,C++	AELINKETH_STATUS AELINKETH_API AelEth_Jog(AELINKETH ael, BYTE addr, int dir);
	C#	public static AethResult AelEth_Jog(System.IntPtr ael, byte addr, int dir);
	VB.NET	Public Shared Function AelEth_Jog(ByVal ael As System.IntPtr, ByVal addr As Byte, ByVal dir As Integer) As Int32
引数	ael	初期化済みハンドル
	addr	対象の AE-LINK アドレス
	dir	回転方向
戻り値		正常に完了した場合、AETH_SUCCESS が返ります。 異常の場合はエラーコード一覧を参照してください。

解説

addr で指定スレーブに対して、コマンド 11h を送信します。
スレーブの連続回転を指令する旭エンジニアリング製の共通コマンドですが、指令後の挙動についてはスレーブに依存します。また、スレーブに対して指令を送受信後 CPU 制御が戻ります。停止指令は別途行ってください。
送信 AE-LINK コマンドは下記の通りです

byte 数	送信パケット
0	0x04
1	addr
2	0x11
3	dir
4	サムチェック

１８－４．AelEth_Homing()

機能		スレーブに対して原点復帰指令を行います。
宣言	C,C++	AELINKETH_STATUS AELINKETH_API AelEth_Homing(AELINKETH ael, BYTE addr);
	C#	public static AethResult AelEth_Homing(System.IntPtr ael, byte addr);
	VB.NET	Public Shared Function AelEth_Homing(ByVal ael As System.IntPtr, ByVal addr As Byte) As Int32
引数	ael	初期化済みハンドル
	addr	対象の AE-LINK アドレス
戻り値		正常に完了した場合、AETH_SUCCESS が返ります。 異常の場合はエラーコード一覧を参照してください。

解説

addr で指定スレーブに対して、コマンド 10h を送信します。
スレーブの原点復帰動作開始を指令する旭エンジニアリング製の共通コマンドですが、指令後の挙動についてはスレーブに依存します。また、スレーブに対して指令を送受信後 CPU 制御が戻ります。停止判定は別途行ってください。
送信 AE-LINK コマンドは下記の通りです

byte 数	送信パケット
0	0x04
1	addr
2	0x10
3	サムチェック

１８－５．AelEth_Relative()

機能		スレーブに対して相対位置決め指令を行います。
宣言	C,C++	AELINKETH_STATUS AELINKETH_API AelEth_Relative(AELINKETH ael, BYTE addr, INT32 mov);
	C#	public static AethResult AelEth_Relative(System.IntPtr ael, byte addr, Int32 mov);
	VB.NET	Public Shared Function AelEth_Relative(ByVal ael As System.IntPtr, ByVal addr As Byte, ByVal mov As Int32) As Int32
引数	ael	初期化済みハンドル
	addr	対象の AE-LINK アドレス
	mov	動作量
戻り値		正常に完了した場合、AETH_SUCCESS が返ります。 異常の場合はエラーコード一覧を参照してください。

解説

addr で指定スレーブに対して、コマンド 12h を送信します。
スレーブの相対位置決め開始を指令する旭エンジニアリング製の共通コマンドですが、指令後の挙動についてはスレーブに依存します。また、スレーブに対して指令を送受信後 CPU 制御が戻ります。停止判定は別途行ってください。
送信 AE-LINK コマンドは下記の通りです

byte 数	送信パケット
0	0x04
1	addr
2	0x12
3	(mov >> 0) & 0xFF
4	(mov >> 8) & 0xFF
5	(mov >> 16) & 0xFF
6	(mov >> 24) & 0xFF
7	サムチェック

１８－６．AelEth_Absolute()

機能		スレーブに対して絶対位置決め指令を行います。
宣言	C,C++	AELINKETH_STATUS AELINKETH_API AelEth_Absolute(AELINKETH ael, BYTE addr, INT32 pos);
	C#	public static AethResult AelEth_Absolute(System.IntPtr ael, byte addr, Int32 pos);
	VB.NET	Public Shared Function AelEth_Absolute(ByVal ael As System.IntPtr, ByVal addr As Byte, ByVal pos As Int32) As Int32
引数	ael	初期化済みハンドル
	addr	対象の AE-LINK アドレス
	mov	位置決め位置
戻り値		正常に完了した場合、AETH_SUCCESS が返ります。 異常の場合はエラーコード一覧を参照してください。

解説

addr で指定スレーブに対して、コマンド 13h を送信します。

スレーブの絶対位置決め開始を指令する旭エンジニアリング製の共通コマンドですが、指令後の挙動についてはスレーブに依存します。また、スレーブに対して指令を送受信後 CPU 制御が戻ります。停止判定は別途行ってください。

送信 AE-LINK コマンドは下記の通りです

byte 数	送信パケット
0	0x04
1	addr
2	0x13
3	(pos >> 0) & 0xFF
4	(pos >> 8) & 0xFF
5	(pos >> 16) & 0xFF
6	(pos >> 24) & 0xFF
7	サムチェック

１８－７．AelEth_DStop()

機能		スレーブに対して減速停止指令を行います。
宣言	C,C++	AELINKETH_STATUS AELINKETH_API AelEth_DStop(AELINKETH ael, BYTE addr);
	C#	public static AethResult AelEth_DStop(System.IntPtr ael, byte addr);
	VB.NET	Public Shared Function AelEth_DStop(ByVal ael As System.IntPtr, ByVal addr As Byte) As Int32
引数	ael	初期化済みハンドル
	addr	対象の AE-LINK アドレス
戻り値		正常に完了した場合、AETH_SUCCESS が返ります。 異常の場合はエラーコード一覧を参照してください。

解説

addr で指定スレーブに対して、コマンド 16h を送信します。
スレーブの原点復帰動作開始を指令する旭エンジニアリング製の共通コマンドですが、指令後の挙動についてはスレーブに依存します。また、スレーブに対して指令を送受信後 CPU 制御が戻ります。停止判定は別途行ってください。
送信 AE-LINK コマンドは下記の通りです

byte 数	送信パケット
0	0x04
1	addr
2	0x16
3	サムチェック

１８－８．AelEth_QStop()

機能		スレーブに対して即停止指令を行います。
宣言	C,C++	AELINKETH_STATUS AELINKETH_API AelEth_QStop(AELINKETH ael, BYTE addr);
	C#	public static AethResult AelEth_QStop(System.IntPtr ael, byte addr);
	VB.NET	Public Shared Function AelEth_QStop(ByVal ael As System.IntPtr, ByVal addr As Byte) As Int32
引数	ael	初期化済みハンドル
	addr	対象の AE-LINK アドレス
戻り値		正常に完了した場合、AETH_SUCCESS が返ります。 異常の場合はエラーコード一覧を参照してください。

解説

addr で指定スレーブに対して、コマンド 16h を送信します。
スレーブの原点復帰動作開始を指令する旭エンジニアリング製の共通コマンドですが、指令後の挙動についてはスレーブに依存します。また、スレーブに対して指令を送受信後 CPU 制御が戻ります。停止判定は別途行ってください。
送信 AE-LINK コマンドは下記の通りです

byte 数	送信パケット
0	0x04
1	addr
2	0x17
3	サムチェック

１８－９．AelEth_Position()

機能		スレーブに対して自動送信データを元に現在位置を取得します。
宣言	C,C++	INT32 AELINKETH_API AelEth_Position(AELINKETH ael, BYTE addr);
	C#	public static Int32 AelEth_Position(System.IntPtr ael, byte addr);
	VB.NET	Public Shared Function AelEth_Position(ByVal ael As System.IntPtr, ByVal addr As Byte) As Int32
引数	ael	初期化済みハンドル
	addr	対象の AE-LINK アドレス
戻り値		正常に完了した場合、AETH_SUCCESS が返ります。 異常の場合はエラーコード一覧を参照してください。

解説

自動送信で取得したスレーブの現在位置を取得します。
なお、本関数を使用するためには、AelEth_AutoAEDrvEntry()で登録し、
AelEth_AutoStart()にて、自動送信を実行状態にする必要があります。

１８－１０．AelEth_Status()

機能		自動送信データを元にステータスを取得します。
宣言	C,C++	UINT32 AELINKETH_API AelEth_Status(AELINKETH ael, BYTE addr);
	C#	public static UInt32 AelEth_Status(System.IntPtr ael, byte addr);
	VB.NET	Public Shared Function AelEth_Status(ByVal ael As System.IntPtr, ByVal addr As Byte) As UInt32
引数	ael	初期化済みハンドル
	addr	対象の AE-LINK アドレス
戻り値		0～ 7bit に通信ステータス、8～15bit に機器ステータスを返します。

解説

自動送信で取得したスレーブのステータスを取得します。
なお、本関数を使用するためには、AelEth_AutoAEDrvEntry()で登録し、
AelEth_AutoStart()にて、自動送信を実行状態にする必要があります。

１８－１１．AelEth_IsMoveing()

機能		自動送信データを元に動作状態を判定します。
宣言	C,C++	BOOL AELINKETH_API AelEth_IsMoveing(AELINKETH ael, BYTE addr);
	C#	public static bool AelEth_IsMoveing(System.IntPtr ael, byte addr)
	VB.NET	Public Shared Function AelEth_IsMoveing(ByVal ael As System.IntPtr, ByVal addr As Byte) As Boolean
引数	ael	初期化済みハンドル
	addr	対象の AE-LINK アドレス
戻り値		動作中　：TRUE 停止中　：FALSE

解説

自動送信で取得したスレーブの動作状態を取得します。
なお、本関数を使用するためには、AelEth_AutoAEDrvEntry()で登録し、
AelEth_AutoStart()にて、自動送信を実行状態にする必要があります。

判定は下記の通りです。

動作中

通信ステータス bit0 又は 機器ステータス bit0 が ON の場合

停止中

通信ステータス bit0 及び 機器ステータス bit0 がともに OFF の場合

１８－１２．AelEth_IsAlram()

機能		自動送信データを元にアラーム状態を判定します。
宣言	C,C++	BOOL AELINKETH_API AelEth_IsAlram(AELINKETH ael, BYTE addr);
	C#	public static bool AelEth_IsAlram(System.IntPtr ael, byte addr)
	VB.NET	Public Shared Function AelEth_IsAlram(ByVal ael As System.IntPtr, ByVal addr As Byte) As Boolean
引数	ael	初期化済みハンドル：
	addr	対象の AE-LINK アドレス
戻り値		アラーム中　　：TRUE アラームでない：FALSE

解説

自動送信で取得したスレーブのアラーム状態を取得します。
なお、本関数を使用するためには、AelEth_AutoAEDrvEntry()で登録し、
AelEth_AutoStart()にて、自動送信を実行状態にする必要があります。

判定は下記の通りです。

アラーム中：

通信ステータス bit2 が ON の場合

アラームでない：

通信ステータス bit2 が OFF の場合

●本資料は、製品をご購入していただくための参考資料となっております。本資料中に記載の技術情報について旭エンジニアリングが所有する知的財産権その他の権利の実施、使用を許諾するものではありません。

●本資料に記載した情報に起因する損害、第三者所有の権利に対する侵害に関し、旭エンジニアリングは責任を負いません。

●本資料に記載した情報は本資料発行時点のものであり、旭エンジニアリングは、予告なしに、本資料に記載した製品または仕様を変更することがあります。

●本資料に記載した情報は正確を期すため、慎重に制作したものですが、万一本資料の記述誤りに起因する損害がお客様に生じた場合には、旭エンジニアリングはその責任を負いません。

●本資料に記載された製品は一般的な産業機器の組込用として設計・製造されています。医療用機器・原子力関係・その他直接人命に関わる機器等には使用しないでください。

●本資料に関し詳細についてのお問い合わせ、その他お気付きの点がございましたら旭エンジニアリング、販売店までご照会ください。

■製造：　株式会社 旭エンジニアリング

小平事業所　〒187-0043　東京都小平市学園東町 3-3-22
Tel：042-342-4422（代）、042-342-4421（技術部・営業部）
Fax：042-342-4423
ホームページ：　http://www.asahi-engineering.co.jp/
Mail：　ae-info@asahi-engineering.co.jp

２０１９年　１２月２４日　改訂